Transport Layer Protocols Lab: TCP/UDP

Part I: Objective

The goal of this lab is to give students an opportunity to observe and analyze the operation of the two most common transport protocols widely used today: TCP-Transport Control Protocol (RFC793) and UDP-User Datagram Protocol (RFC768). This will help clarify and re-enforce some of the concepts students have come across in readings and in class lectures. This lab will help you to understand:

TCP connection establishment and termination.
TCP window-based flow control.
TCP retransmission.
TCP congestion control.
TCP window management: slow start.
Differences between TCP and UDP.

Part II: Equipment List

Students will use the Ethereal network analyzer to capture, observe and analyze frames from FTP transfers and also trace files which have been pre-captured. The following is the equipment needed.

1. One PC.
2. The Ethereal network analyzer software installed.
3. Trace files:
   Tcp1.cap
   Tcp2.cap
   Udp.cap

Part III: Introduction and Background

TCP provides a reliable stream-oriented end-to-end transport service. Bytes presented to TCP are transported error free and in sequence from one point to the other. TCP treats data passed from higher layer entities as an unstructured, continuous byte stream to be delivered transparently from end-to-end. Any concept of structure or organization to the stream is the responsibility of the upper layers. TCP provides reliability through acknowledgement, timeout and retransmission. Window-based flow control, adaptive timers and other techniques allow TCP to respond to network congestion efficiently.

UDP provides an un-reliable end-to-end datagram based transport service. Each message passed to UDP for delivery is treated as a separate entity and is offered best-effort service. Essentially, UDP appends port numbers and an optional UDP header checksum to add end-to-end service on top of IP. UDP does not implement any flow control, error control or congestion control. Data may be delivered out of sequence, incomplete, in error, or not at all.

Part IV: Lab Procedures

1. Go to the PC labeled "cuckoo"
2. Log in with the user name : **tele2310** and password : **networks**
3. From the windows desktop, you will see the **Ethereal** network analyzer shortcut.
4. Run the Ethereal network analyzer by double clicking on the Ethereal icon
5. From the menu bar, go to Capture and select Start. A small Capture window will show up, and then click OK.
6. Run FTP (double click on FTP shortcut) to get a file from your SIS or CS account. Choose a rather small file (1 – 10 Kbytes) for this transfer.
7. Wait until FTP transfers file completed.
8. Logout from the ftp server
9. Select Stop on Ethereal's capture window.
10. The screen will now show all frames captured. It includes frames from FTP and other frames in the network.
11. To observe your FTP frames only, you have to do frame filtering. At the bottom of the screen, type in **ip.addr eq <*cuckoo's ip address*>**, and hit Enter. (You can find cuckoo's ip address posted on the machine)
12. The screen will now display all your FTP captured frames on the screen.
13. From the menu bar, go to Display and select Expand All

At this point, take a moment to get familiar with the display. The display screen comprises 3 panes: summary, details and hex. When you move a mouse to any frame/field, the display will highlight the corresponding frame/field in every pane. You can look around the display; try to see TCP fields such as source port, destination port, sequence number, acknowledgement number, flags, frame size, etc.

1. TCP Connection Establishment

TCP is a connection oriented transport layer protocol. A specific sequence of action is required to establish a connection. This sequence of actions is called the 3-way handshake.

From frames you captured, identify the frames, which are included in the connection establishment phase. For each frame, identify the following information and record in the table 1 below.

No. The number of frames assigned by the network analyzer.
Src. Source name or IP address.
Dest. Destination name or IP address.
Size. Size of the frame.
Src Port. TCP source port. (You can record only last 6 digits of all 10 digits.)
Dest Port. TCP destination port. (You can record only last 6 digits of all 10 digits.)
Flag. Record the set flag such as ACK, SYN.
Seq. Sequence number.
Ack. Acknowledgement number.
Win. TCP advertised window.

No	Src	Dest	Size	Src Port	Dest Port	Flag	Seq	Ack	Win

Table 1 TCP connection establishment.

Questions

1.1 Which flag does TCP use to establish a connection? What is its purpose?
1.2 Which field of TCP defines the maximum segment size that will be accepted on this connection? What is the maximum segment size?

2. TCP Connection Termination

Similar to the connection establishment phase, identify the frames, which are included in the connection termination phase. Record the frame information in the table 2 below.

No	Src	Dest	Size	Src Port	Dest Port	Flag	Seq	Ack	Win

Table 2 TCP connection termination.

Questions

2.1 How does the TCP terminate its connection?

2.2 How does the other TCP confirm the termination?

3. TCP Window-based Flow Control

The window-based flow control used by TCP is known as a credit allocation scheme. When a TCP sender sends a frame, it includes the sequence number of the first octet in the segment data field. A TCP receiver acknowledges an incoming frame with a message of the form (A = *i*, W = *j*), with the following interpretation:

- All octets through sequence number *i* - 1 are acknowledged; the next expected octet has sequence number *i*.
- Permission is granted to send an additional window (W) of *j* octets of data; that is, the *j* octets corresponding to sequence numbers *i* through *i* + *j* –1.

Note that the receiver is not required to immediately acknowledge incoming segments, but may wait and issue a cumulative acknowledgement for a number of segments.

Next, you will observe a pre-captured trace file.

1. If there's a filter defined in the Filter box, hit the clear button next to the box to clear the effects of any visualization filter
2. From the menu bar, go to File and select Open.
3. Select the file "Tcp1.cap" from the C:\tcplab\ directory. This file is a trace of FTP from one station to the other. It a half duplex exchange that you can easily observe frames.

Observe how TCP uses flow control in the data exchange phase. Jump to frame 112 (from the menu bar, go to the Edit menu, select Go To Frame and type in the frame number) and record the frame information in the table 3 below.

No	Src	Dest	Size	Src Port	Dest Port	Flag	Seq	Ack	Win
112							X	X	X
113									
114							X	X	X
115							X	X	X
116									
117							X	X	X
118							X	X	X
119									
120							X	X	X
121							X	X	X
122									
123									
124							X	X	X
125							X	X	X

Table 3 TCP window-based flow control.

Questions

3.1 How many segments and how many bytes of incoming frames will the receiver acknowledge the sender?

3.2 List frame numbers, which the receiver acknowledges the sender without granting additional credit? Why does the receiver do that?

3.3 List frame numbers, which the receiver acknowledges the sender and increases more credit? How many credits are increased? How does the receiver do that?
3.4 Why does the sending TCP set the PUSH flag on each segment?

4. TCP Retransmission

In case of lost segments, TCP does not use any negative acknowledgement such as NAK or REJ. TCP utilizes positive acknowledgement, timeout and retransmission to ensure error-free, sequenced delivery of segments. If the retransmission timer expires before an acknowledgement is received, the data is retransmitted starting at the byte after the last acknowledgement byte in the stream.

From the trace file, you will see a scenario in which a segment is lost during the course of a transmission stream. The sender does not know how many bytes were in the lost segment and continues to transmit segments to the receiver. Until the retransmission timer at the sender expires, it will resend unacknowledged data from the stream.

Observe how TCP handles retransmissions and record the frame information in table 4. Jump to frame 37 and, from the Display menu, select Options and select Seconds since beginning of capture.

No	Time (since beginning of capture)	Src	Dest	Size	Flag	Seq	Ack	Win
37							X	X
38							X	X
39								
40								
41							X	X
42								
43								
44							X	X
45							X	X
46								
47								
48							X	X
49							X	X
50								
51								
52							X	X
53							X	X
54								
55								
56							X	X
57							X	X
58								
59								
60							X	X
61							X	X
62								
63								
64							X	X
65								
66								
67							X	X

Table 4 TCP retransmission.

Questions
4.1. Which frame number is the lost segment? What is its sequence number?
4.2. How long is the retransmission timeout?
4.3. The TCP standard provides several possible implementation options, for instance, accept policy: in order or in window. From what you have observed, does the receiver accept incoming segments before it receives the retransmitted segment? Which accept option does the receiver implement?
4.4. How does the receiver inform the sender that it have accepted all subsequent segments since the lost segment?

5. TCP Congestion Control

The way that TCP deals with congestion is known as self-clocking behavior. TCP can automatically sense the network bottleneck and regulate its flow accordingly. The returning acknowledgement functions as a pacing signal. In steady state, the sender's segment sending rate will match the rate of the acknowledgement. Therefore, the sending rate will equal that of the slowest part on the network, which could be the network itself or the receiver.

In this part, you will have a chance to identify the bottleneck by observing the captured frames of a transmission. Jump to frame 114 and, from the Display menu, select Options and select Seconds since previous frame. Record the frame information in the table 5 below.

No	**Time** (since previous frame)	**Src**	**Dest**	**Size**	**Flag**	**Seq**	**Ack**	**Win**
114							X	X
115							X	X
116								
117							X	X
118							X	X
119								
120							X	X
121							X	X
122								

123								
124							X	X
125							X	X

Table 5 TCP congestion control.

Questions

5.1. Look at the time between frames in the table 5; which frame number implies the bottleneck? Is it the network or the receiver?

5.2. What is the cause of bottleneck?

6. TCP Window Management: Slow Start

You have learned from the previous observation that TCP has a self-clocking behavior, which can control congestion when a bottleneck is present. However, when a connection is initialized, the sending TCP has no idea how many segments it can transmit in order to not cause congestion. If the number of segments is too large, it might cause more congestion. One technique TCP uses to handle this problem is slow start.

Slow start makes uses of a congestion window, measured in segments instead of octets. The sending system will transmit segments based on the following:

awnd = min[credit, cwnd]

where:

awnd = allowed window, in segments, which the sending TCP can transmit.
cwnd = congestion window, in segments. It is a window that TCP uses in the startup period and during congestion.
credit = the advertised window from the recent acknowledgement, in segments, which equals window/segment size.

Briefly, when a connection is open, the sending TCP initializes cwnd = 1. When a segment is transmitted and the sender receives an acknowledgement, the value of cwnd is increased by 1. Thus, the sender can now send 2 segments. Each time the sender gets an acknowledgement; it will be increase cwnd by 1.

For this part of the lab, you will observe the slow start behavior when TCP opens a connection. Open the trace file "Tcp2.cap" and record the frame information table 6. You need to set Options to Seconds since beginning of capture.

No	Time (since beginning of capture)	Src	Dest	Size	Flag	Seq	Ack	Win
4								
5								
6								
7								
8								
9								
10								
11								
12								
13								
14								
15								
16								
17								
18								
19								
20								
21								
22								
23								
24								
25								
26								
27								

Table 6 TCP window management: slow start.

Questions

6.1. Start from frame number 4, select only frames sent by the source and plot a graph between sequence number (Y-axis) and time (X-axis)?

6.2. How many is the maximum segments that the sender can transmit?

7. Differences Between TCP and UDP

As mentioned previously, UDP provides an un-reliable, best-effort, end-to-end datagram based transport service. In this part of the lab you will observe some UDP frames and see differences with TCP.

Open the file "Udp.cap" and answer the following questions.

Questions

7.1 What are fields in the UDP header?
7.2 What is the difference between TCP connection establishment and what UDP does?
7.3 How would UDP deal with lost frames?
7.4 What functionality does UDP add to IP?